Time Constraints in Mixed Multi-unit Combinatorial Auctions

Andreas Witzel[1]* and Ulle Endriss[2]

[1] CIMS, New York University, USA
[2] ILLC, University of Amsterdam, Netherlands

Abstract. We extend the framework of mixed multi-unit combinatorial auctions to include time constraints, present an expressive bidding language, and show how to solve the winner determination problem for such auctions using integer programming. Mixed multi-unit combinatorial auctions are auctions where bidders can offer combinations of transformations of goods rather than just simple goods. This model has great potential for applications in the context of supply chain formation, which is further enhanced by the integration of time constraints. We consider different kinds of time constraints: they may be based on either time points or intervals, they may determine a relative ordering of transformations, they may relate transformations to absolute time points, and they may constrain the duration of transformations.

1 Introduction

Cerquides et al. [2] have proposed an extension of the standard combinatorial auction model, called **mixed multi-unit combinatorial auctions** (or simply **mixed auctions**). In a mixed auction, bidders can offer transformations, consisting of a set of input goods and a set of output goods, rather than just plain goods. Bidding for such a transformation means declaring that one is willing to deliver the specified output goods after having received the input goods, for the price specified by the bid. Solving a mixed auction means choosing a sequence of transformations that satisfies the constraints encoded by the bids, that produces the goods required by the auctioneer from those he holds initially, and that maximizes the amount of money collected from the bidders (or minimizes the amount paid out by the auctioneer). Mixed auctions extend several other types of combinatorial auctions: direct auctions, reverse auctions, and combinatorial exchanges. A promising application is supply chain formation.

We propose extending the framework of mixed auctions by allowing bidders to specify constraints regarding the times at which they perform the transformations offered in their bids. The motivation for this extension is that, in a complex economy, the bidders (service providers) themselves may need services from others and have their own supply chains, so the bidders may have preferences over the timing of transformations and over their relative ordering. A notion of time is already implicit in the original framework as

* Most work was done at ILLC on a GLoRiClass fellowship funded by the European Commission (Early Stage Research Training Mono-Host Fellowship MEST-CT-2005-020841).

far as the auctioneer is concerned, who builds a sequence of transformations, but this is not the case for the bidders. In this work we seek to redress this imbalance.

Our contribution covers four types of **time constraints**:

- *Relative time points:* associate each transformation with a time point and allow bidders to express constraints regarding their relative ordering, e.g., transformation X must be executed before Y.
- *Absolute time points:* additionally allow references to absolute time, e.g., execute X at time 15, or at most 3 time units after Y.
- *Intervals:* associate transformations with intervals and specify constraints, e.g., X must be executed *during* Y.
- *Intervals with absolute durations:* allow intervals with absolute time, e.g., X should take at least 5 time units.

These constraint types can be freely mixed to, for instance, express an interval taking place after a time point. Furthermore, it is possible to model *soft constraints*, allowing bidders to offer discounts in return for satisfying certain time constraints, and to model the fact that an auctioneer may sometimes be able to quantify the monetary benefit resulting from a shorter supply chain. Our approach blends nicely into the existing framework of mixed auctions, requiring surprisingly few modifications. This facilitates integration with other extensions and optimizations.

In Sect. 2, we define a suitable **bidding language**. In Sect. 3, we define the **winner determination problem** and present an integer program to solve it. Section 4 presents the extension to time intervals. Section 5 discusses related work and concludes.

2 Bidding language

In a mixed auction, agents negotiate over transformations of goods that are equipped with time point identifiers. In this section we introduce an expressive bidding language that allows bidders to specify their valuations over sets of such transformations. We also present some purely syntactic extensions to the bidding language, and we show that it is fully expressive over the class of all "reasonable" valuations.

2.1 Transformations and time points

Let G be the finite set of all types of goods considered. A **transformation** is a pair $(\mathcal{I}, \mathcal{O}) \in \mathbb{N}^G \times \mathbb{N}^G$. An agent offering such a transformation declares that, when provided with the multiset of goods $\mathcal{I}$, he can deliver the multiset of goods $\mathcal{O}$. Let $\mathcal{T}$ be a finite (but big enough) set of **time point identifiers**. These time points are merely identifiers, not variables having an actual value. They can be referred to from bids in order to specify time constraints over the offered transformations. Agents negotiate over sets of transformations with time point identifiers $\mathcal{D} \subset \mathbb{N}^G \times \mathbb{N}^G \times \mathcal{T}$, which we can write as

$$\mathcal{D} = \{(\mathcal{I}^1, \mathcal{O}^1, \tau^1), \ldots, (\mathcal{I}^\ell, \mathcal{O}^\ell, \tau^\ell)\}.$$

For example, $\{(\{\}, \{q\}, \tau_1), (\{r\}, \{s\}, \tau_2)\}$ means that an agent is able to deliver q without any input at some time τ_1, and to deliver s if provided with r at some time τ_2 (possibly with constraints regarding τ_1 and τ_2).

2.2 Valuations

A **time line** Σ (for a given bidder) is a finite sequence of transformations and "clock ticks" c (when no transformation is allocated to the bidder). That is, $\Sigma \in (\mathbb{N}^G \times \mathbb{N}^G \cup \{c\})^*$. A **valuation** v maps a time line Σ to a real number p. Intuitively, $v(\Sigma) = p$ means that an agent with valuation v is willing to make a payment of p for getting the task of performing transformations according to the time line Σ (p is usually negative, so the agent is *being paid*). We write $v(\Sigma) = \bot$ if v is *undefined* for Σ, i.e., the agent would be unable to accept the corresponding deal. For example, the valuation v given by

$$\begin{aligned} v((\{oven, dough\}, \{oven, cake\})) &= -2 \\ v((\{oven, dough\}, \{oven, cake\}); (\{\}, \{bread\})) &= -3 \\ v((\{\}, \{bread\}); (\{oven, dough\}, \{oven, cake\})) &= \bot \end{aligned}$$

expresses that for two dollars I could produce a cake if given an oven and dough, also returning the oven; for another dollar I could do the same and afterwards give you a bread without any input; but I could not do it the other way round.

A valuation v uses **relative time** if for all Σ we have that $v(\Sigma) = v(\Sigma - c)$, where $\Sigma - c$ stands for Σ with all clock ticks c removed. That is, valuations depend only on the relative ordering of the transformations. Otherwise v is said to use **absolute time**.

2.3 Bids

An **atomic bid** $\textsc{bid}(\mathcal{D}, p)$ specifies a finite set of finite transformations with time points and a price. For **complex bids**, we restrict ourselves to the XOR-*language*, which, for mixed auctions, fully expresses most (if not all) intuitively sensible valuations [2]. Our framework can easily be extended to also handle the OR-operator. An XOR-bid,

$$\textit{Bid} = \textsc{bid}(\mathcal{D}_1, p_1) \text{ XOR} \ldots \text{XOR } \textsc{bid}(\mathcal{D}_n, p_n),$$

says that the bidder is willing to perform *at most one* $\mathcal{D}_j$ and pay the associated p_j.

2.4 Time constraints

The **atomic constraints** for *relative time* are of the form $\tau < \tau'$; and for *absolute time*, with $\tau, \tau' \in \mathcal{T}$, $\xi, \xi' \in \mathbb{N}$:

$$\tau = \xi \qquad \tau < \xi \qquad \tau > \xi \qquad \tau + \xi < \tau' + \xi' \qquad \tau + \xi = \tau' + \xi'$$

As mentioned above, the τ are not variables but just identifiers for the associated transformations, and the above formulas are not assignments but rather constraints on the associated transformations, with semantics as given in Sect. 2.5. For example,

$$\begin{aligned} \textsc{bid}(\{\ &(\{oven, dough\}, \{oven, cake\}, \tau_1), \\ &(\{\}, \{bread\}, \tau_2)\}, -3) \end{aligned} \qquad \tau_1 < \tau_2$$

expresses the above fact that I am willing to sell you the bread only *after* the cake.

Time **constraint formulas** are of the form $\varphi = \gamma_1 \wedge \cdots \wedge \gamma_\nu$ with atomic constraints γ_ι. A bidder submits a bid *Bid* together with a time constraint formula φ, expressing that he is willing to perform according to *Bid*, but *only under the condition* that φ is satisfied. This condition is *hard*: the bidder will only accept if it is met.

2.5 Semantics

Syntactically, we thus have *complex bids with time points* together with *constraint formulas* over these time points:

$$\text{BID}(\mathcal{D}_1, p_1) \text{ XOR} \dots \text{XOR BID}(\mathcal{D}_n, p_n) \qquad\qquad \gamma_1 \wedge \dots \wedge \gamma_\nu.$$

In order to make the intuitive meanings explained above explicit, we now specify a formal semantics. In the following, let Σ be a time line (clock ticks allowed), let $\tau, \tau' \in \mathcal{T}$, $\xi, \xi' \in \mathbb{N}$, and let φ and φ' be time constraint formulas. Let $\tau \in \Sigma$ denote the fact that τ is associated with some transformation in Σ, and let $\Sigma(\tau)$ denote the sequence number (starting from 1) of the transformation associated with τ, if $\tau \in \Sigma$. For clarity, we may include the time point identifiers in the sequence. For example, if $\Sigma = ((\mathcal{I}^1, \mathcal{O}^1, \tau^1); \dots)$, then $\tau^1 \in \Sigma$ and $\Sigma(\tau^1) = 1$.

We inductively define a **satisfaction relation** $\models$ as follows:

$$\begin{array}{ll}
\Sigma \models \tau \circ \xi & \text{iff } \tau \notin \Sigma \text{ or } \Sigma(\tau) \circ \xi, \text{ for } \circ \in \{=, <, >\} \\
\Sigma \models \tau + \xi < \tau' + \xi' & \text{iff } \tau' \notin \Sigma \text{ or} \\
 & \quad \tau \in \Sigma \text{ and } \Sigma(\tau) + \xi < \Sigma(\tau') + \xi' \\
\Sigma \models \varphi \wedge \varphi' & \text{iff } \Sigma \models \varphi \text{ and } \Sigma \models \varphi'
\end{array}$$

Relative time constraints are covered by omitting the $+\xi$ and $+\xi'$, and $\tau + \xi = \tau' + \xi'$ is an abbreviation for

$$\tau + \xi < \tau' + (\xi' + 1) \wedge \tau' + \xi' < \tau + (\xi + 1).$$

According to this semantics, time constraints over time point identifiers that are fully included in Σ are interpreted as expected. Constraints over time point identifiers not in Σ are simply ignored (they are always satisfied). Note that the choice of semantics for constraints such as $\tau < \tau'$ is somewhat arbitrary in case only one of the time points being compared occurs in Σ. As an intuitive justification for this detail of the semantics, τ may be thought of as a precondition for τ', for instance, because some outcome of the first transformation is needed for the second. In the case of $\tau + \xi = \tau' + \xi'$, this has the effect that either none of the two mentioned transformations is included, or both are and must have the specified distance. However, the exact details do not matter all that much, since the bidding language allows specifying in all detail which transformations can occur together and which cannot.

Using a more technical justification, we prefer this interpretation of constraints because it turns out that it has a straightforward translation to integer constraints, which we need for the implementation described in Sect. 3.3.

We say that a set of transformations $\mathcal{D}$ **permits** Σ if Σ consists of *exactly* the transformations in $\mathcal{D}$ (and optionally clock ticks). In contrast to this definition, in [2], different assumptions concerning *free disposal* are distinguished. Informally, free disposal means that participants are always happy to accept *more* goods than they strictly require; if they really have absolutely no use for them (or are even bothered by them), they can dispose of them *for free*. Free disposal makes intuitive sense for most every-day goods; however it is not as appropriate for certain "goods" like nuclear waste. We do not delve further into this issue here and continue without any free-disposal assumptions; however, we emphasize that this is purely for the sake of clarity, and these assumptions could be built in with only minuscule changes. In particular, the issue of free disposal as far as bidders are concerned has no impact on the winner determination problem discussed in Sect. 3; it only affects the definition of the semantics of the bidding language.

We now define the valuation expressed by an atomic bid $\mathit{Bid} = (\mathcal{D}, p)$ together with a time constraint formula φ as

$$v_{\mathit{Bid},\varphi}(\Sigma) = \begin{cases} p & \text{if } \mathcal{D} \text{ permits } \Sigma \text{ and } \Sigma \models \varphi \\ \bot & \text{otherwise.} \end{cases}$$

Accordingly, the valuation expressed by a complex bid $\mathit{Bid} = \text{XOR}_{j=1}^{n} \mathit{Bid}_j$ together with a time constraint formula φ is (interpreting $\bot$ as $-\infty$):

$$v_{\mathit{Bid},\varphi}(\Sigma) = \max\{v_{\mathit{Bid}_j,\varphi}(\Sigma) \mid j \in \{1, \dots, n\}\}.$$

That is, out of all the applicable atomic bids Bid_j (i.e., where $v_{\mathit{Bid}_j,\varphi}(\Sigma) \neq \bot$), the auctioneer is allowed to choose the one giving him maximum profit.

2.6 Syntactic extensions

The time constraint language may seem limited, allowing only conjunctions of atomic constraints. However, additional expressive power can be "borrowed" from the bidding language. We discuss two extensions to the time constraint language.

Disjunctive time constraints. A bidder may want to offer $(\mathcal{I}^1, \mathcal{O}^1)$, $(\mathcal{I}^2, \mathcal{O}^2)$ and $(\mathcal{I}^3, \mathcal{O}^3)$ for price p, where the third should take place after the second *or* the first, i.e.,

$$\text{BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \tau^1), (\mathcal{I}^2, \mathcal{O}^2, \tau^2), (\mathcal{I}^3, \mathcal{O}^3, \tau^3)\}, p) \qquad \tau^1 < \tau^3 \lor \tau^2 < \tau^3,$$

with the obvious meaning of the disjunction $\lor$. This is not directly possible in our time constraint language. However, it can be translated into

$$\begin{array}{l} \text{BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \vartheta^1), (\mathcal{I}^2, \mathcal{O}^2, \vartheta^2), (\mathcal{I}^3, \mathcal{O}^3, \vartheta^3)\}, p) \\ \quad \text{XOR BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \zeta^1), (\mathcal{I}^2, \mathcal{O}^2, \zeta^2), (\mathcal{I}^3, \mathcal{O}^3, \zeta^3)\}, p) \end{array} \quad \vartheta^1 < \vartheta^3 \land \zeta^2 < \zeta^3.$$

The choice which of the disjuncts to satisfy has been moved into the bid expression and is determined by picking one of the atomic bids. Since their variables are disjoint, this pick makes one conjunct of the transformed time constraint formula vacuously true, while the other conjunct still needs to be satisfied. Since it may happen that both of the original disjuncts are satisfied in the end, disjunction is the right notion here, even though it is translated into an XOR of bids.

For a general formulation, we allow a bid expression in XOR normal form together with a time constraint formula in disjunctive normal form:

$$\text{XOR}_{j=1}^{n} \mathit{Bid}_j \qquad\qquad \bigvee\nolimits_{\iota=1}^{\nu} \varphi_\iota,$$

where the φ_ι are standard (conjunctive) time constraint formulas. The bidder can thus conveniently express, e.g., several alternative partial orders over his transformations.

Let now σ_ι for $\iota \in \{1, \dots, \nu\}$ be substitutions (with disjoint ranges), each mapping all variables occurring in the bid to fresh (used nowhere else) ones. The translation is:

$$\text{XOR}_{\iota=1}^{\nu}\, \text{XOR}_{j=1}^{n}\, \mathit{Bid}_j \sigma_\iota \qquad\qquad \bigwedge_{\iota=1}^{\nu} \varphi_\iota \sigma_\iota .$$

This may seem surprising, because in the original formulation the auctioneer has two choices (which of the time constraint disjuncts to satisfy and which bid to pick), and in the translation he loses the choice among the time constraints. However, in return he gets the freedom to choose over the outer XOR. As illustrated in the example above, this boils down to choosing one of the fresh variable spaces, which corresponds to choosing one of the original disjuncts. All the rest of the transformed time conjunction does not have any effect, because it talks about variables which do not occur in the chosen sub-bid. The auctioneer then proceeds to pick a bid from the inner XOR, just as before.

Soft time constraints. Soft constraints are constraints with associated costs. Intuitively, such a constraint does not have to be satisfied, but if it is, the price of the bid is modified by the given cost (usually a discount to the auctioneer).

For example, a bidder may want to bid on $(\mathcal{I}^1, \mathcal{O}^1)$ and $(\mathcal{I}^2, \mathcal{O}^2)$ for price p and offer a discount, i.e., raise his bid by δ, if he gets to do the first before the second:

$$\text{BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \tau^1), (\mathcal{I}^2, \mathcal{O}^2, \tau^2)\}, p) \qquad\qquad (\tau^1 < \tau^2, \delta).$$

Again, this expression can be translated:

$$\begin{array}{ll} \text{BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \vartheta^1), (\mathcal{I}^2, \mathcal{O}^2, \vartheta^2)\}, p) & \\ \quad \text{XOR BID}(\{(\mathcal{I}^1, \mathcal{O}^1, \zeta^1), (\mathcal{I}^2, \mathcal{O}^2, \zeta^2)\}, p + \delta) & \qquad \zeta^1 < \zeta^2. \end{array}$$

The general translation is analogous to the previous one and omitted for space reasons.

Note also that the transformations can be combined. For example, a soft time constraint could have a disjunctive condition.

The **blowup** resulting from the transformations is straightforwardly seen to be linear in the number of disjuncts or of alternative discounts, respectively. Our constructions and the resulting blowup straightforwardly carry over to any bidding language that allows XOR as outermost connective.

2.7 Expressive power

We say a valuation is **finite** if it has a finite domain (i.e., yields non-$\perp$ for finitely many time lines only) consisting of finite sequences of finite transformations (i.e., with finite input and output). The XOR language with time constraints is **fully expressive** for finite valuations: XOR bids with *relative* time constraints can express all finite valuations that use relative time; XOR bids with *absolute* time constraints can express all finite valuations. The proof is simple: Take an XOR bid with one atomic bid $\text{BID}(\mathcal{D}, p)$ for each Σ in the domain of v, with $\mathcal{D}$ set to permit Σ and p set to $v(\Sigma)$, and impose the order corresponding to Σ using time constraints (note that there may be several atomic bids with the same transformations in $\mathcal{D}$, but different time points).

3 Winner determination

We now study the winner determination problem (WDP). This is the problem, faced by the auctioneer, of determining which transformations to award to which bidder, so as to maximize (minimize) the sum of payments collected (made), given the bids of the bidders expressed in our bidding language. This may be interpreted as computing a solution that maximizes revenue for the auctioneer, or utilitarian social welfare for the collective of bidders (if we interpret prices offered as reflecting bidder utility). Note that we are interested in the *algorithmic* aspects of the WDP. Game-theoretical considerations, such as how to devise a more sophisticated pricing rule that would induce bidders to bid truthfully, are orthogonal to the algorithmic problem addressed here. (We briefly comment on mechanism design issues in Sect. 5, but this is not the topic of this paper.)

For symmetry between bidders and auctioneer, we do *not* assume free disposal for the auctioneer (just like for the bidders), i.e., he does not want to end up with any goods except the required ones. Note, however, that the formulations are easily adapted to allow free disposal (and we point out the necessary changes along the way).

After formulating the WDP, we give an integer program [9] solving it. We aim at keeping the descriptions short and focus on the changes compared to the version from [2]. The advantage of this approach, besides showing how few modifications are necessary, is that it is modular and can (hopefully) be combined without too much effort with other extensions or optimizations.

3.1 WDP with time constraints

The *input* to the WDP consists of

- a bid expression *Bid*$_i$ in XOR normal form together with a conjunction of time constraints φ_i, for each bidder i;
- a multiset $\mathcal{U}_{in}$ of goods the auctioneer holds in the beginning;
- and a multiset $\mathcal{U}_{out}$ of goods the auctioneer wants to end up with.

Let *Bid*$_{ij}$ denote the jth atomic bid BID$(\mathcal{D}_{ij}, p_{ij})$ occurring within *Bid*$_i$, let t_{ijk} be a unique label for the kth transformation in $\mathcal{D}_{ij}$ (for some arbitrary but fixed ordering of $\mathcal{D}_{ij}$), and let τ_{ijk} be the time point identifier associated with transformation t_{ijk}. Let $(\mathcal{I}_{ijk}, \mathcal{O}_{ijk})$ be the actual transformation labelled with t_{ijk}, and T be the set of all t_{ijk}.

An **allocation sequence** Σ resembles the time line we introduced before, but can only contain transformations actually offered by some bidder, and each one at most once. That is, Σ now is a permutation of a subset of T, possibly interspersed with clock ticks c.

We write $t_{ijk} \in \Sigma$ to say that the kth transformation in the jth atomic bid of bidder i has been selected, and we write $\Sigma(t_{ijk})$ to denote the sequence number of t_{ijk} (starting from 1) if $t_{ijk} \in \Sigma$. By Σ_i we denote the **projection** of Σ to bidder i, that is, Σ with each t_{ijk} replaced by $(\mathcal{I}_{ijk}, \mathcal{O}_{ijk}, \tau_{ijk})$ and all $t_{i'jk}$ replaced by c for $i' \neq i$. By $(\mathcal{I}^m, \mathcal{O}^m)$ we denote the mth transformation in Σ. Thus, we have two ways of referring to a selected transformation: by its position in the received bids (t_{ijk}) and by its position m in the allocation sequence.

Given Σ, we can inductively define the bundle of goods held by the auctioneer after each step (let $g \in G$ be any good, and let $\mathcal{M}^0 = \mathcal{U}_{in}$):[3]

$$\mathcal{M}^m(g) = \mathcal{M}^{m-1}(g) + \mathcal{O}^m(g) - \mathcal{I}^m(g) \tag{1}$$

under the condition that

$$\mathcal{M}^{m-1}(g) \geq \mathcal{I}^m(g). \tag{2}$$

Given a multiset $\mathcal{U}_{in}$ of goods available to the auctioneer, a multiset $\mathcal{U}_{out}$ of goods required by the auctioneer, and a set of bids Bid_i with time constraints φ_i, an allocation sequence Σ is a **valid solution** if

(i) for each bidder i, some $\mathcal{D}_{ij}$ permits Σ_i, or $\Sigma_i \in \{c\}^*$
(ii) for each bidder i, $\Sigma_i \models \varphi_i$
(iii) Eq. (1) and (2) hold for each transformation $(\mathcal{I}^m, \mathcal{O}^m) \in \Sigma$ and each good $g \in G$
(iv) for each good $g \in G$, $\mathcal{M}^{|\Sigma|}(g) = \mathcal{U}_{out}(g)$.[4]

The **revenue** for the auctioneer associated with a valid solution Σ is the sum of the prices of the selected atomic bids, i.e., $\sum \dot{\{} p_{ij} \mid \exists k : t_{ijk} \in \Sigma \dot{\}}$.

Given multisets $\mathcal{U}_{in}$ and $\mathcal{U}_{out}$ of initial and required goods and a set of bids with time constraints, the **winner determination problem** (WDP) consists in finding a valid solution that maximizes the auctioneer's revenue.

3.2 Original integer program

In this part, we closely follow Cerquides et al. [2]. The main issue is to decide, for each offered transformation, whether it should be selected for the solution sequence, and if so, at which position. Thus, we define a set of binary decision variables $x_{ijk}^m \in \{0, 1\}$, each of which takes on value 1 if and only if the transformation t_{ijk} is selected at the mth position of the solution sequence.

The position number m ranges from 1 to an upper bound M on the solution sequence length. For the time being, we take $M = |T|$, the overall number of transformations, accommodating all sequences that can be formed using only transformations (and not clock ticks). We consider an alternative way for specifying M at the end of Sect. 3.3.

Further, i ranges over all bidders; j ranges for each bidder i from 1 to the number of atomic bids submitted by i; and k ranges for each atomic bid j of bidder i from 1 to the number of transformations in that bid.

We use the following auxiliary binary decision variables: x^m takes on value 1 if and only if any transformation is selected at the mth position; x_{ijk} takes on value 1 if and only if transformation t_{ijk} is selected at all; and x_{ij} takes on value 1 if and only if any of the transformations in the jth atomic bid of bidder i are selected.

The following set of constraints define a valid solution *without* taking time constraints into account (i.e., neglecting (ii) in the valid solution definition above):

(1) Select either all or no transformations from an atomic bid (cf. (i) above):

$$x_{ij} = x_{ijk} \quad (\forall ijk)$$

[3] Given a multiset $\mathcal{S} \in \mathbb{N}^G$ and an item $g \in G$, we write $\mathcal{S}(g)$ to denote the number of copies of g in $\mathcal{S}$.

[4] Replace $=$ by $\geq$ to model free disposal.

(2) Select at most one atomic bid from each XOR normal form bid (cf. (i) above):
$\sum_j x_{ij} \leq 1 \quad (\forall i)$

(3) Select each transformation at most for one position: $x_{ijk} = \sum_m x^m_{ijk} \quad (\forall ijk)$

(4) For each position, select at most one transformation: $x^m = \sum_{ijk} x^m_{ijk} \quad (\forall m)$

(5) There should be no gaps in the sequence: $x^m \geq x^{m+1} \quad (\forall m)$
Note that this is strictly speaking not required; indeed we drop this constraint later on in order to allow clock ticks between transformations.

(6) Treating each $\mathcal{M}^m(g)$ as integer decision variable, ensure that necessary input goods are available (cf. (iii) above):

$$\mathcal{M}^m(g) = \mathcal{U}_{in}(g) + \sum_{\ell=1}^{m} \sum_{ijk} x^\ell_{ijk} \cdot (\mathcal{O}_{ijk}(g) - \mathcal{I}_{ijk}(g))$$
$$\mathcal{M}^m(g) \geq \sum_{ijk} x^m_{ijk} \cdot \mathcal{I}_{ijk}(g) \qquad (\forall g \in G, \forall m)$$

(7) In the end, the auctioneer should have the bundle $\mathcal{U}_{out}$ (cf. (iv) above):[5]
$\mathcal{M}^M(g) = \mathcal{U}_{out}(g) \quad (\forall g \in G)$

Solving the WDP now amounts to solving the following integer program:

$$\max \sum_{ij} x_{ij} \cdot p_{ij}, \quad \text{subject to constraints (1)–(7)}$$

A valid solution is then obtained by making transformation t_{ijk} the mth element of the solution sequence Σ exactly when $x^m_{ijk} = 1$.

3.3 Modified integer program

To implement time constraint handling (thus obeying (ii) in the definition of valid solution given above), we first introduce an additional set of auxiliary binary decision variables $y^m_{ijk} \in \{0, 1\}$, taking on value 1 if and only if transformation t_{ijk} is selected at the mth position *or earlier* in the solution sequence. This is achieved by adding the following constraint:

(8) y^m_{ijk} should be 1 iff $t_{ijk} \in \Sigma$ and $\Sigma(t_{ijk}) \leq m$: $y^m_{ijk} = y^{m-1}_{ijk} + x^m_{ijk} \quad (\forall ijkm)$, with $y^0_{ijk} = 0$.

We now give implementations for our two variants of time constraints.

Relative time. Each bidder i's time constraint formula is a conjunction of atomic time constraints, and all bidders' time constraints need to be satisfied. The following set of integer constraints takes care of this.

(9a) For each $\tau_{ijk} < \tau_{ij'k'}$ occurring in $\bigwedge_i \varphi_i$: $y^m_{ijk} \geq y^{m+1}_{ij'k'} \quad (\forall m)$.

In accordance with the time constraint semantics, if neither t_{ijk} nor $t_{ij'k'}$ occurs in the solution sequence, this requirement is vacuously satisfied since both sides stay 0. If $t_{ij'k'}$ *does* occur, then $y^m_{ij'k'}$ will become 1 at some point m. In this case, the requirement boils down to y^{m-1}_{ijk} being 1 as well, so t_{ijk} must have occurred already.

Solving the WDP with relative time constraints thus amounts to the same optimization as before, but subject to constraints (1)–(8) and (9a).

Absolute time. In order to have an absolute notion of time, we need some way of mapping points of a possible solution sequence to an absolute time line. The simplest

[5] With free disposal, $=$ would become $\geq$.

way is to interpret each sequence point itself as a time unit (a minute, a day, a week, ...), and this is the approach we take.

Before giving the formalization, we need to discuss some conceptual details. If we interpret steps in the sequence as absolute time units, some issues arise which did not matter before. Firstly, while it may be acceptable to break time down into discrete steps of equal duration, it is not so easy to defend that any transformation that can possibly be offered should have exactly that duration. Secondly, there is no reason why the auctioneer should wait for one transformation to end before commissioning the next transformation, which may be offered by a different, idling bidder, unless the output of the former is needed as input to the latter. To some extent, these issues can be addressed by a purely conceptual extension presented in Sect. 4. However, we leave it to future work to design frameworks which handle time in a more flexible way and truly optimize for effective parallelizations. For our purposes, we simply assume that the auctioneer is busy when he is delivering or receiving goods of some particular transformation, and cannot deal with several bidders simultaneously.

To start the formalization, first of all we drop constraint (5). As mentioned, it is not strictly speaking necessary anyway, and since now the bidders can refer to arbitrary absolute time points, we actually might have to accept gaps in the sequence.

Now a technical issue arises: The length of possible solution sequences is no longer bounded by $|T|$. While it may be possible to find a correct bound by looking at all numbers occurring in the bidders' time constraints, we settle for a different solution: The auctioneer manually specifies M, the maximum length of the solution sequence.

At first glance this seems like a pure loss of generality; however the auctioneer may profit from having some control over the size of the WDP he has to solve, and he can always iterate over different values for M in his search for a good solution. Economically speaking, it also makes sense that the auctioneer wants some control over the length of his supply chain, rather than allowing an arbitrary length. Indeed, he might have graded preferences over the time his supply chain takes, as discussed in Sect. 3.4.

We now give the integer constraints for handling absolute time constraints.

(9b) For each $\tau_{ijk} + \xi < \tau_{ij'k'} + \xi'$ occurring in $\bigwedge_i \varphi_i$: $\quad y_{ijk}^{m+\xi'} \geq y_{ij'k'}^{m+\xi+1} \quad (\forall m)$

For each $\tau_{ijk} + \xi = \tau_{ij'k'} + \xi'$ occurring in $\bigwedge_i \varphi_i$: $\quad y_{ijk}^{m+\xi'} = y_{ij'k'}^{m+\xi} \quad (\forall m)$

(10) For each $\tau_{ijk} \circ \xi$, with $\circ \in \{=, <, >\}$, occurring in $\bigwedge_i \varphi_i$: $x_{ijk}^{m} = 0 \quad (\forall m \not\circ \xi)$.

Constraint (9b) requires some explanation. First of all, note that (9a), the version for relative time, is covered as a special case. As indicated by the semantics, the absolute time variant is thus an extension of the relative time variant. Secondly, note that the second half of (9b) can be obtained from the first half if interpreted as an abbreviation, as in Sect. 2.5. Now consider the case where $\xi' = 0$. Intuitively speaking, the time constraint then says that t_{ijk} must take place at least $\xi + 1$ time steps before $t_{ij'k'}$. That is, whenever $t_{ij'k'}$ is selected, t_{ijk} must already have been selected for at least $\xi + 1$ time steps. In terms of the integer program, this means that, for all positions m, $y_{ij'k'}^{m}$ must be 0 *unless* $y_{ijk}^{m-\xi-1}$ was already 1. Now it is only a small step to the formulation in (9b).

Solving the WDP with absolute time constraints amounts to the same optimization as before, but subject to constraints (1)–(4), (6)–(8), (9b) and (10).

A valid solution is then obtained by making transformation t_{ijk} the mth element of the solution sequence Σ if and only if $x^m_{ijk} = 1$, and using a clock tick c as mth element when there is no x^m_{ijk} which equals 1 (i.e., when $x^m = 0$).

3.4 Valuation for the auctioneer

Given that we decided to require the auctioneer to specify the maximum length M of the solution sequence (for the absolute-time variant of the framework), we may also want to enable him to express more detailed preferences over durations. This can be achieved in a neat way, also enabling the auctioneer to express graded preferences over final bundles.

So assume the auctioneer derives a certain value from a given supply chain, depending on its overall duration and its outcome, the bundle of goods he owns in the end. Note that we here assume absolute time; with relative time, preferences over *durations* do not make much sense, but the results can easily be adjusted to only model preferences over *outcomes*.

We thus assume the auctioneer's valuation is a function $u : \mathbb{N} \times \mathbb{N}^G \to \mathbb{R} \cup \{\bot\}$, mapping duration/outcome pairs to a value or $\bot$, meaning the duration/outcome pair is not acceptable. This valuation can be incorporated into the WDP in the following way.

After receiving the bids, the auctioneer decides on a maximum duration M and creates an additional bid under an unused bidder identity:

$$\underset{\{(m,\mathcal{U}) \mid u(m,\mathcal{U}) \neq \bot\}}{\text{XOR}} \text{BID}(\{(\mathcal{U}, \{\circledcirc\}, \tau_{m,\mathcal{U}})\}, u(m,\mathcal{U})),$$

where $\circledcirc$ is a special token that does not occur as a good in any other bid, together with time constraints:

$$\bigwedge_{\{(m,\mathcal{U}) \mid u(m,\mathcal{U}) \neq \bot\}} \tau_{m,\mathcal{U}} = m.$$

The transformations in this bid are to be thought of as terminal transformations: they denote the possible time points and outcomes at which a solution sequence may end, and the associated values for the auctioneer. Using this method, the auctioneer's valuation can be expressed with very few changes to the integer program:

– The terminal transformations should only be used at the respective intended positions in the sequence; this is ensured by the given time constraints.
– At *most* one of them should be used; this is ensured by the XOR (and strictly speaking also follows from the last point below).[6]
– At *least* one[7] of them should be used; this can be ensured by setting $\mathcal{U}_{out} = \{\circledcirc\}$.
– The unique terminal transformation that is actually used should indeed be the end of the solution sequence.[8]

[6] Even more strictly speaking, it also follows from the next requirement and the fact that we assume no free disposal; we include it nevertheless for conceptual clarity and in order to accommodate a possible free disposal assumption.

[7] This could read "*exactly* one", but again, we want to accommodate a possible free disposal assumption.

[8] As a last remark, this requirement could be dropped if we *did* assume free disposal and all bids' prices were positive.

For this last point, we need an additional integer constraint:

(11) No transformations are scheduled after a terminal transformation:

$$x_{ijk}^{m+1} \leq 1 - y_{-1j'k'}^{m} \quad (\forall ijkj'k'm)$$

(-1 being the auctioneer's "bidder identity").

While the above requirements could also be encoded more directly and more efficiently into the integer program, for clarity we here restrict ourselves to this version using the high-level features of the bidding language.

Many further extensions and optimizations along these lines are conceivable. We do not try to exhaust them here, but sketch only one example. The auctioneer might want to extract some goods $\mathcal{U}$ from the supply chain by some intermediate time point ξ, not necessarily at its end. To express this, he can add a transformation $(\mathcal{U}, \{\diamond\}, \tau)$ with time constraint $\tau < \xi + 1$ to his bid, and add $\diamond$ to $\mathcal{U}_{out}$. Dropping constraint (11) for this transformation, he makes it non-terminal. He can also make this a soft requirement by including another transformation that yields $\diamond$ from no input and attaching appropriate prices to the corresponding bids.

3.5 Computational complexity

As in the original model [2], the WDP for mixed auctions with time constraints is NP-complete: NP-hardness follows from NP-hardness of the WDP for standard combinatorial auctions [8] and NP-membership follows from the fact that the validity of a given allocation sequence can clearly be verified in polynomial time. Fionda and Greco [4] have started to chart the tractability frontier for a slightly simplified version of the original framework by Cerquides et al. [2], using various criteria to restrict the class of allowed bids. Their results concerning the XOR language still hold in our extended framework. In particular, even with time constraints the WDP still remains tractable if only one transformation outputs any particular good [4, Theorem 3.7]. Further tractability islands may be identified in future work.

Regarding the integer program, while there is room for optimizations, the number of variables we introduce is of the same order as in the original formulation: $O(n^2)$, where n is the number of transformations occurring in the bids submitted. The most recent work on winner determination algorithms for mixed auctions has tried to reduce the number of decision variables needed so as to improve performance [5,7]. Due to the modular nature of our approach, we are optimistic that it will be possible to take advantage of these optimizations and integrate them with our extensions.

4 Intervals

It may be desirable to allow transformations to overlap or take place during others, and to allow transformations to have different durations. Interestingly, **intervals** can be handled in our framework without any additional machinery. A transformation with start time and end time can be rewritten into two transformations with single time points and an appropriate time constraint:

$$(\mathcal{I}, \mathcal{O}, [\tau, \tau']) \quad \rightsquigarrow \quad \begin{array}{l} (\mathcal{I}, \emptyset, \tau), (\emptyset, \mathcal{O}, \tau') \\ \tau < \tau' \end{array}$$

Since the replacement takes place within a single atomic bid, it is guaranteed that either both the start and end transformations will be selected, or neither. That is, the interval transformation remains intact.

The usual interval relations (see the interval calculus by Allen [1]; due to sequentiality we consider only strict relations) can be defined as macros:

$$\begin{aligned}
[\tau_1, \tau_1'] \text{ BEFORE } [\tau_2, \tau_2'] &\rightsquigarrow \tau_1' < \tau_2 \\
[\tau_1, \tau_1'] \text{ OVERLAPS } [\tau_2, \tau_2'] &\rightsquigarrow \tau_1 < \tau_2 \wedge \tau_1' < \tau_2' \\
[\tau_1, \tau_1'] \text{ DURING } [\tau_2, \tau_2'] &\rightsquigarrow \tau_2 < \tau_1 \wedge \tau_1' < \tau_2
\end{aligned}$$

With absolute time, absolute restrictions on the durations can also be implemented:

$$duration([\tau, \tau']) \circ \xi \rightsquigarrow \tau' \circ \tau + \xi, \qquad \circ \in \{<, >, =\}$$

Note that expressions like $duration(\cdot) > duration(\cdot)$ are not so straightforwardly expressible, but arguably also much less useful in the context of specifying bids.

5 Conclusions and related work

We presented an extension to the existing framework of mixed multi-unit combinatorial auctions [2], enabling bidders to impose time constraints on the transformations they offer.

In the original framework, the auctioneer is free to schedule the offered transformations in any way suitable to achieve his desired outcome, while bidders are left with no control over this process. Our work redresses this asymmetry, thus representing an important step towards a more realistic model of supply chain formation, where bidders themselves may have supply chains or other factors restricting the possible schedules for performing certain transformations.

Starting from a very basic core language for expressing time constraints, we have given various extensions, many purely syntactic, showing the somewhat unexpected power inherent to the core language.

We have also extended the integer program given in [2] to handle time constraints. Our extensions are modular in a way that will facilitate combining them with other extensions and optimizations for mixed auctions, such as [5,7].

Time constraints have been applied to different types of combinatorial auctions in the literature. For example, Hunsberger and Grosz [6] extended an existing algorithm for winner determination in combinatorial auctions to allow precedence constraints when bidding on roles in a prescribed action plan ("recipe"). Collins [3] permitted relative time constraints in a combinatorial reverse auction over combinations of tasks, and tested the efficiency of various approaches to solving the winner determination problem.

Auction frameworks involving time have also been fruitfully applied to problems of distributed scheduling. In Wellman et al. [10], time constraints do not enter separately, rather time slots are the actual objects being auctioned, and game-theoretic and mechanism design issues are discussed.

While it would be interesting to examine whether the insights about efficiency and alternative approaches to handling time could be applied to our framework, the roles, tasks, and time slots being auctioned in those contributions are "atomic", and the formulations and results do not easily translate to transformations in the context of mixed auctions.

Concerning mechanism design, with finite valuations, the incentive-compatibility of the Vickrey-Clarke-Groves (VCG) mechanism carries over from standard combinatorial auctions to mixed multi-unit combinatorial auctions with time constraints (see also [2]). It is a question of independent interest, whether and how this can be extended to non-finite valuations when still allowing only finite bids. In this case the bidders would not be able to express their true valuations exactly, so it is not obvious how truthfulness and incentive compatibility should be defined.

Other topics for future work include the exact interplay between the various syntactic extensions we have given, defining a uniform general language, and determining whether some of the features can be implemented in more direct (and efficient) ways than through the translation to the core language used in this work. The same holds for the underlying bidding language, where operators such as OR may be executed more efficiently than through translation to XOR.

Changing the integer program to allow OR instead of XOR is straightforward; an XOR-of-OR language, generalizing both the XOR and the OR languages, can be accommodated with more extensive changes, buying the advantage of preserving our constructions for disjunctive and soft time constraints.

Finally, an empirical analysis needs to be performed, including testing and optimizing our integer program.

References

1. J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
2. J. Cerquides, U. Endriss, A. Giovannucci, and J. A. Rodríguez-Aguilar. Bidding languages and winner determination for mixed multi-unit combinatorial auctions. In *Proc. IJCAI-2007*, Hyderabad, India, 2007.
3. J. E. Collins. *Solving combinatorial auctions with temporal constraints in economic agents*. PhD thesis, University of Minnesota, 2002.
4. V. Fionda and G. Greco. Charting the tractability frontier of mixed Multi-Unit combinatorial auctions. In *Proc. IJCAI-2009*, Pasadena, CA, July 2009.
5. A. Giovannucci, M. Vinyals, J. A. Rodríguez-Aguilar, and J. Cerquides. Computationally efficient winner determination for mixed multi-unit combinatorial auctions. In *Proc. AAMAS-2008*. IFAAMAS, 2008.
6. L. Hunsberger and B. J. Grosz. A combinatorial auction for collaborative planning. In *Proc. 4th International Conference on MultiAgent Systems*. IEEE Computer Society, 2000.
7. B. Ottens and U. Endriss. Comparing winner determination algorithms for mixed multi-unit combinatorial auctions. In *Proc. AAMAS-2008*. IFAAMAS, 2008.
8. M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinational auctions. *Management Science*, 44(8):1131–1147, Aug. 1998.
9. A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.
10. M. P. Wellman, W. E. Walsh, P. R. Wurman, and J. K. MacKie-Mason. Auction protocols for decentralized scheduling. *Games and Economic Behavior*, 35(1–2):271–303, 2001.